The Dolphin Site Manager's Guide

A guide to keeping your
Xerox 1100 Scientific Information Processor
Happy

Eric Schoen[1]
SUMEX-AIM Computer Project
Stanford University

June 1982

This work was funded by the NIH Biotechnology Resources Program under grant RR-00785. Material in the Appendices I and III is adapted from Xerox internal memos; permission to reproduce them here is gratefully acknowledged.

---

[1]The author's current address is: Schlumberger-Doll Research, P.O. Box 307, Ridgefield, CT 06877

Table of Contents

Chapter One
Introduction

The Xerox 1100 Scientific Information Processor (better known as a *Dolphin*, or sometimes, *D0*) is a personal workstation capable of emulating a number of target architectures, such as the Xerox Alto,[2] and the Mesa and Interlisp Virtual Machines. The Dolphin is equipped with a 23 megabyte Winchester disk for local storage, a 1024 by 808 bitmapped display, a 3 megabit Ethernet interface and at least 2304 512-byte pages of MOS memory (with a large virtual address space).

This manual will discuss the responsibilities of the site manager for a Dolphin installation running the Interlisp-D language. Interlisp-D is the implementation of the Interlisp language for the Xerox's D-series machines (this includes the Dolphin, the Dorado, and future processors). Interlisp-D is fully compatible with Interlisp-10 and Interlisp-VAX, but has the added power of bitmapped display graphics not available in these other Interlisp implementations.

In addition, the Interlisp-D I/O system makes full use of the capabilities of Ethernet file transfer protocols. Files can reside either on the local disk or on a file server accessible via the Ethernet. Paged and random-access I/O can be performed on both local and remote files.

Other References

This guide provides only a cursory overview of the Alto operating system, and does not attempt to describe Interlisp at all. For more complete information on the Alto system, refer to *The Alto User's Handbook*. Interlisp is documented in *The Interlisp Reference Manual*. Those parts of Interlisp-D not defined in the basic Interlisp language are documented in the *Interlisp-D Users Guide*. An excellent introduction to the LISP language is found in *LISP*, by Patrick Winston and Berthold Klaus Paul Horn. The 3 mbit/sec Ethernet is documented in a number of Xerox internal memos, as well as a small number of published papers. Xerox should be able to supply copies of these papers.

---

[2] The Alto is an early Xerox personal computer, based on the Data General Nova. It has a 600 by 800 point bitmapped display, a local disk drive, and a connection to the experimental 3 Megabit Ethernet. The Alto served as the basis for all facets of Xerox's personal workstation development including processor, software, and perhaps most importantly, Ethernet development. The Altos are comparatively slow machines, but they are still around, and the software developed for them is still in use in machines which emulate Altos.

Chapter Two
The Dolphin Hardware

The configuration of a Dolphin running Interlisp-D typically contains:

- The processor, containing ALU, control store, memory controller, miscellaneous functions, memory, disk, terminal, and Ethernet interface boards.
- The local disk, currently a Shugart 4008 Winchester-style disk.
- The keyboard, display, and mouse. The display is a 1024 by 808 bitmap arranged in landscape (horizontal) mode. The mouse is a pointing device to aid in interactive graphical programming.

Dolphins come in three styles depending on their vintage of production. The original Dolphins were built at Xerox PARC and are encased in Alto boxes. These are small units with the Shugart disk built-in, and invisible from the outside. The "pre-production" Dolphins were built at Xerox's El Segundo plant and are actually the front end processors for their 5700 series laser printers. These are larger, louder machines, with the Shugart disk placed in its own box sitting on top of the processor box. Finally, the production model Dolphins are once again small units (smaller than the PARC models) with carefully designed acoustical padding to make them acceptable in an office setting.

Despite their varied appearances, however, all Dolphins are electrically and operationally compatible. Most PC boards and all programs can be swapped between units.

The processor box

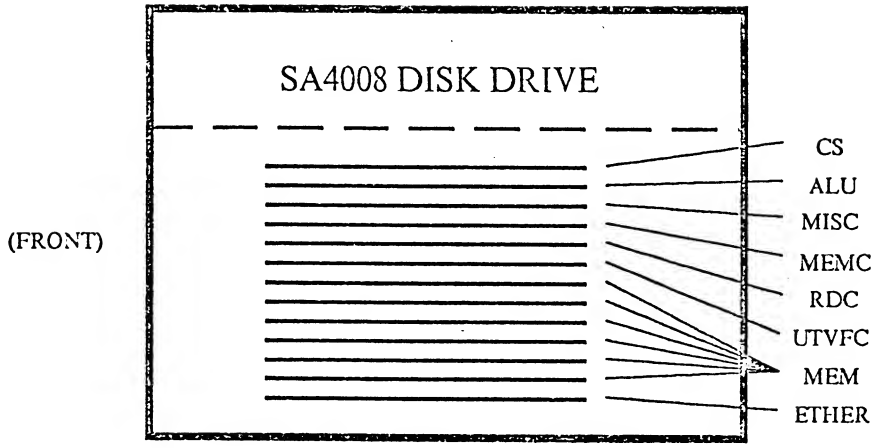
The Dolphin's central processor is composed of four boards: the control store, ALU, miscellaneous functions, and memory controller boards. These boards always occupy the first four slots in the motherboard.[3] See figure 2-1. Furthermore, each board has a specific slot within these four locations, and is keyed to prevent its being installed in the wrong slot. The rest of the PC boards can go almost anywhere in the motherboard (memory boards must reside in slots 5-12) so long as no gaps exist between boards.

The front panel of the processor box contains the ON/BOOT and OFF switches, as well as a 4 LED maintenance panel. The ON/BOOT switch turns the processor on, and later (on a second push) boots the processor, once the disk has warmed up. The maintenance panel (MP) displays various information about the state of the processor. The meanings of the various numbers are detailed in Appendix III. When the processor is first booted, the MP displays the number of pages of physical memory available. While running Interlisp, the MP displays a cumulative count of page faults since starting Interlisp.[4]

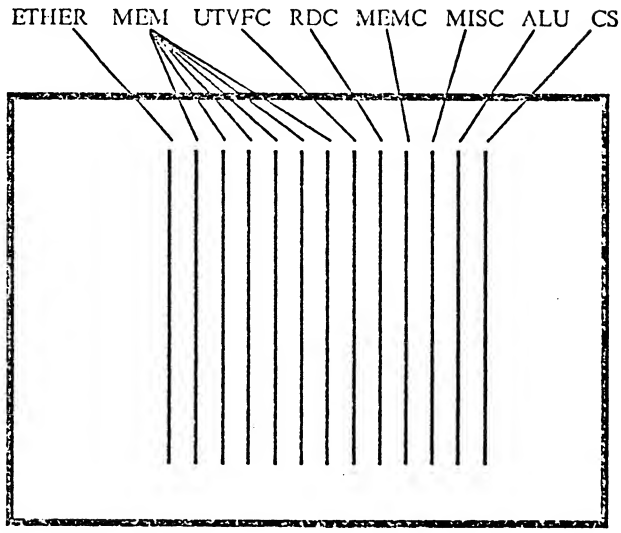
---

[3] Far right on the pre-production model, looking into the back of the unit; top-to-bottom on the production units.

[4] This may have changed by the time you read this document.



PRODUCTION DOLPHIN (SIDE VIEW)



PRE-PRODUCTION DOLPHIN (REAR VIEW)

Figure 2-1: Backplane configuration for pre-production and production Dolphins

In the following, references to the tops and bottoms of the PC boards are meaningful for the pre-production model of the Dolphin, in which the boards are mounted vertically from the rear of the processor box.

The rigid disk controller (RDC) board allows the Shugart disk to emulate a dual model 44 Diablo disk. This permits software developed for the Alto to be used with the Dolphin. There are two cables linking the RDC board with the disk. The cable terminated in the small plug is plugged into the upper of the two small sockets on the rear of the board.

The terminal controller board (UTVFC in Xerox terminology) links the keyboard and display with the processor. The cable to the display should be plugged into the topmost of the four sockets on the board. The display can be as far as 250 feet from the processor.

There are usually six memory boards, each containing 96K 16-bit words of memory. When the Dolphin is booted, the maintenance panel on the front of the Dolphin records the number of pages of memory available. This should read at least 2304. If less than that (frequently in quanta of 384 pages, the capacity of one memory board), at least one memory board is failing. Interlisp-D can usually run with 1920 pages of memory (one memory board failing), but its performance will be poor. The Dolphin can be configured with a maximum of eight memory boards. As you might expect, its performance will be better with more physical memory.

The Ethernet Interface connects the Dolphin to a 3 Megabit Ethernet. There is a single cable leading from the interface to the Ethernet transceiver. This cable can be about 40 feet long.

A typical procedure for tracking down hardware failures is to swap PC boards with a working Dolphin. Make sure the processor is powered off when changing boards; severe component damage will result if you do not.

The Keyboard, Display, and Mouse

The keyboard/display/mouse station is linked to the processor via a seven twisted-pair cable. As previously noted, this cable can be up to 250 feet long; however, at these distances, the screen image may be corrupted by line noise. If you attempt to remote the display by this much, you must adhere to the cable specifications as provided by Xerox; a low-loss, high bandwidth cable is required. Be sure you have not crossed the twisted pairs; to do so sometimes produces quite a pyrotechnic display behind the screen!

If you remote the display unit, you may have to adjust a timing delay circuit located inside the display base. To do this, remove the back of the display base. Attached to the back is a small PC board labelled "7-Wire Interface." Sitting on top of the PC board is an ungainly LRC circuit. This circuit must be adjusted to match the characteristics of the cable between the display and the processor. If misadjusted, the display will appear to fragment. Adjust the variable capacitor until the screen image appears whole again. Once you reassemble the display, you may find the adjustment of the LRC is off slightly due to its being installed within the base; if this happens, try again.

The mouse is a useful device which permits the user to point to any location on the screen. The mouse currently in use has a large steel ball on its underside. Motions of the mouse across a surface are translated into X and Y components and then transmitted to the processor. The mouse performs best if placed on an unpolished surface. Paper is generally not a good surface to run the mouse on, though, since paper fibers can jam up the precision mechanism inside the mouse. The best surface is a thick rubber mat (Xerox calls them "mouse traps").

The keyboard is a 63 key unencoded keypad. Because the keyboard is unencoded, the Dolphin can read any number of keys simultaneously, and also give arbitrary meaning to any key. Within Interlisp-D, however, the keyboard is treated like a standard CRT keyboard.

Notice the set of three unlabeled keys on the right side of the keypad. The lowest key is known as the SWAT key, and when used in conjunction with the left SHIFT or CONTROL key, allow you to interrupt Interlisp and return to the Alto operating system or enter the Alto debugger respectively. This is not a good idea, as you may destroy the Interlisp environment, but you may need to resort to this in case of software or hardware problems which hang Interlisp.

The Local Disk

The local disk is built into the processor box on the PARC and production model Dolphins, and sits on top of the processor box in the pre-production models. As mentioned above, there are two cables running from the disk to the disk controller in the processor. A third cable connects to a power supply within the processor, as well. The disk should be maintenance free, however, you may need to replace a disk which has developed hard disk errors, rendering it unsafe to use.

Replacing a Disk

To replace a disk, first unplug all cables leading to it. Gain access to the top of the disk (remove the cover of the disk box on the pre-production model). There are two locking devices you must install before moving the disk. The first of these is a head restraint. There should be a plastic "fork" in or near the Dolphin. This should be placed under the head motion damper (the circular fixture at the rear of the disk with the yellow caution labels) around the head pivot to prevent the heads from moving during transport. Do this while the disk is powered up.[5] The second lock is a small screw which locks the belt drive, preventing the disk from spinning. There is a yellow caution label with an arrow pointing to the place where the screw goes (on the black plastic guard surrounding the drive belt). Rotate the drive pulley by hand (clockwise only!) until the screw hole comes into view. Install the screw and tighten until firm.

To remove the disk from its box, unplug all cables leading from the disk fixture to the box, and loosen the screws on the four legs which fix the disk to its case. Remove the disk and put the replacement disk in its place. Connect the necessary plugs, remove the drive locking screw and the head damper lock, and plug the cables from the disk drive into the processor. Put the cover back on. Power the Dolphin up and listen for the usual sounds of the disk drive. If things sounds wrong, double check what you've done, and call Xerox if necessary. If the disk sounds OK, you may still have to reinstall the Interlisp software. See page 15.

---

[5] Yes, powered-up!. Shugart says that while spinning, the head floats above the disk, and it is impossible to damage the head in case inserting the damper restraint causes the head to move.

Chapter Three
The Alto Software

Dolphin software falls into two categories: that which is necessary to run the Alto emulator, and that which is necessary to run Interlisp. A freshly-booted Dolphin runs an Alto emulator. The Alto emulator is necessary because many of the lowest-levels of I/O in the Interlisp environment rely on BCPL routines written for the Alto. When running only the Alto emulator, the display will appear narrow, masked to the dimensions of an Alto screen. The Alto also provides a filesystem. In the Dolphin, your filesystem appears to spread over two separate partitions of a dual Diablo model 44 disk drive. The two disk partitions are isolated from one another; to switch between partitions, give the Partition.~ command (e.g. type **Partition 1** or **Partition 2**).

The Alto command processor (the Alto Executive) has a syntax reminiscent of the Tenex/Tops-20 Exec. Interlisp users should have little use for the Alto emulator, aside from starting Interlisp, but the Dolphin site manager should be familiar with the Alto Executive and operating system, as several diagnostics (notably disk diagnostics) must be invoked using Alto commands.

Alto Executive Commands

Commands to the Alto Executive can be abbreviated, and recognition is available using the ESC key. File names and commands are treated identically by the command processor. There are no special characters in Alto file names; "." has no significance in a file name. However, commands to the Alto Executive are signified by ending in ".~" (i.e. Copy.~). Filenames and commands can be a mixture of upper and lower case; however, all command and filename lookup is done in a case independent manner. When entering commands or file names, several special characters can be typed:

?	Type out file names or commands which fit the partial name/command typed in so far. Return to command/file name input afterwards.
TAB	Like "?," except that the command input buffer is cleared after listing the file names or commands which match what had been typed in. TAB typed to a blank field is the way to see the names of all the files on the disk and all the Executive commands.
↑X	Control-X. Expands any wildcards entered so far, replacing them by all the file names which fit the wildcard mask. For instance, if files FOO1 and FOO2 exist, and you type "DELETE FOO*" followed by ↑X, the command line is replaced by DELETE FOO1 FOO2, allowing you to abort the command in case a file you didn't mean to delete is among the wildcard group.
DEL	Aborts the command, returns to command input. Like ↑U on Tops-20, DEL on vanilla Tenex, ↑X on Sumex Tenex, etc.
BS	Backspaces one character.
ESC	Attempts to recognize and complete input typed so far. If ambiguous, fills out as far as it can.

Alto Executive commands to know:

Copy.~ Copies a file on the disk. The syntax is not Tenex/Tops-20-like. To copy a file, type Copy.~ *NewFileName* ← *OldFileName.*

Delete.~ Deletes files on the disk. Type one or more filenames, separated by spaces: Delete.~ *file1 file2* ...etc... *fileN.*

Install.~ Causes the Alto operating system to enter a reconfiguration dialogue. The "short dialogue" allows you to set a default username, a disk name, and whether a password is needed to use the machine. The disk name should identify which partition of the disk is in use (i.e. call partition one **Partition 1** and partition two **Partition 2**). Dolphin users may desire that a password be set to prevent unauthorized use of the machine. You should know these passwords in case you have to diagnose failures when no one who knows the passwords is around.

The long installation dialogue allows you to erase the disk and completely reinstall the Alto and Interlisp systems. This may have to be done periodically when new releases of Interlisp come out, as well as when you are installing a new disk. The long installation dialogue cannot be performed unless you have another Dolphin accessible over the Ethernet running the Alto FTP program, or unless you have access to a larger file server running programs compatible with the Xerox 3 Megabit PUP software (specifically, the Byte Stream Protocol based File Transfer Protocol). Appendix I contains the long install procedure.

Login.~ Tells the Alto operating system your name and password. Interlisp-D uses this when it attempts to do file transfers over the Ethernet. If you forget to login at the Alto level, there is a LOGIN function in Interlisp to do the same thing. If you are reinitializing Interlisp (see page 11), you should give the Login.~ command first so that Interlisp knows whose INIT.LISP file to look at when running the GREET function.

NetExec.~ If your Dolphins are connected to a 3 Megabit Ethernet which has a file server with net boot capabilities, you can give the NetExec command. This causes the Dolphin to boot the NetExec, a special network-oriented Executive stored on the file server. The NetExec in turn lets you boot other programs from the file server. The NetExec gives you access to diagnostics (DiEx, CRTTest, KeyTest) and filesystem verification and repair programs (Scavenger) you may need to keep your Dolphins running.

Partition.~ Allows you to switch between partitions of the disk.

Rename.~ Renames a file on the disk. The syntax is like Copy.~ (see above).

Scavenger.~ Boots the Scavenger program via the Ethernet. Scavenger is a filesystem repair program which can undo damage caused by a crash. Instructions on its use are in *The Interlisp-D User's Guide.*

Alto Files and Programs

There are a number of files critical to the operation of the Alto emulator and Alto operating system. If any of these files are deleted, you won't be able to boot your Dolphin. This can be an insurmountable problem if you don't have the necessary 3 Megabit software running on your file server to reinitialize the Alto software from scratch.

Files needed for the Alto OS:

DiskDescriptor.	Defines the filesystem format.
Executive.Run	This is the Alto command processor.
Swat.	This is the Alto debugger. A crashed Alto usually ends up in Swat, giving you some indication of what went wrong (i.e. disk error, trap instruction, etc).
Swat.Help	A list of Swat commands, readable by Swat, giving you online help when you type "?."
Sys.Boot	This is the Alto operating system.
Sys.Errors	A text file used by Swat to translate between internal error numbers and human readable messages.
SysFont.al	.AL files are font files for Altos. SysFont.al is the font used by the Executive and other Alto programs.
User.cm	A text file, defining preferences for various user-settable options in the Executive and other Alto programs.

Alto programs come in two flavors, run files and boot files. Run files can be invoked from the Alto Executive. They usually have the .Run extension, and when you quit from such a program, you are returned to the Executive. To invoke a run file, type its name to the Alto Executive, followed by any optional arguments (see the reference to CreateFile.Run, for instance, on page 12).

Boot files, on the other hand, cannot be invoked directly from the Alto Executive (they can on an Alto, but the Dolphin has the wrong type of disk for the Alto Bootfrom.~ command to work). Boot files can be invoked by the NetExec, though (see above). When a program in a boot file quits, it usually returns you to the NetExec program. As their name might suggest, boot files replace the entire Alto operating system. Any information you've told the Alto operating system about yourself (i.e. your name, using the Login.~ command) is lost when you run a boot file. To invoke a boot file, type its name to the NetExec program. You cannot supply arguments with a boot file, however.

In the list below, programs available only as run files or boot files are so indicated by their extension; programs available in both forms are listed without any extensions in their names.

Useful Alto Programs:

Chat	The Alto virtual terminal program. You may have this program (you will if you are at a University Grant site or you've convinced Xerox to let you have it) or you may not. There's also DMChat, which simulates a 60 line high Datamedia 2500 CRT.
Createfile.Run	Creates big empty files using contiguous disk space. This is the way to create Interlisp-D's virtual memory buffer when you are reinstalling the interlisp-D system.

CRTTest.Boot — Useful for checking display linearity. Type any character to switch to the next sized grid.

DiEx.Boot — The Diablo disk exerciser. Run this program if you suspect disk problems. DiEx has many options which can be set by selecting an option with the mouse and entering new values. For most purposes, it suffices to set the disk type to Diablo 44 (point to Type: Mod-31 and press the left mouse button), initialize (point to Init Verify and press the left mouse button), and run the test (point to Do Test and press a mouse button). DiEx will report hard and soft disk errors, giving their sector, track, and head numbers. For more customized diagnosis, look at some of the other menu items in DiEx (they are self-documenting; simply point at an item with the mouse, and read the message in the message window). A more rigorous diagnostic is available in the BFSTest program, documented in Appendix I.

FTP — The Alto BSP based file transfer program. If you have Chat, you'll probably have FTP. If your file server isn't running a reliable Leaf server, it may be running a reliable FTP server (implementations exist for Tenex, Tops-20, and Unix, as well as for the Xerox Alto-based Interim File Server).

InstallSwat.Run — Reinstalls the Alto debugger. You must run this after erasing the disk with the long installation dialogue.

KeyTest.Boot — Useful for checking the keyboard. Self-explanatory.

Scavenger — Scavenger is a file system repair program. When your Dolphin falls into Swat with an error message telling you that there's something wrong with the disk, try running Scavenger. Scavenger takes about 5 minutes to run. When it's done, it leaves a log called Scavenger.Log$ telling you what it's done, and creates a file called Scavenger.Garbage$ which contains "incorrigible disk pages." You may delete Scavenger.Garbage$ with the Delete.~ command. If your disk has hard read errors (caused perhaps by surface damage on the disk), Scavenger will map them out of existence.[6]

The Alto Net Boot Procedure

This information is included for sites with 3 megabit Ethernet software wishing to add or understand the Alto net boot protocol. The Alto net bootstrap does the following:

1. Broadcasts a *MayDay* Pup to all miscellaneous service sockets on the connected network. The ID field of the Pup contains the boot file number of the file to be booted.

2. Becomes an EFTP receiver. A server which receives the *MayDay* request and can satisfy it opens an EFTP socket, and begins sending the boot file to the Alto or Dolphin. If two servers or more can satisfy a boot request, the bootee locks onto the first one which responds, and ignores all others (space limitations within the Alto prevented the inclusion of code to send *Abort* Pups to other responding servers); the other servers will eventually timeout.

Of course, the above description presupposes an Ethernet bootstrap in memory. This may not be the case. It is possible to boot an Alto or Dolphin with a blank memory (i.e. one which has no

---

[6] Sites running a Scavenger predating May 12, 1982 should ask for an update.

working disk to load a bootstrap from) by means of the *Breath-of-Life* mechanism. Every five seconds, a Breath-of-Life server will broadcast a special packet (not a Pup, but a raw Ethernet packet of type 602B, directed to host 377B). An Alto or Dolphin wishing to be booted will copy this packet into main memory, starting at location 1, and begins executing code starting at location 3. The packet may be no longer than 254 words plus 2 words of Ethernet encapsulation. *Breath-of-Life* is a block of Alto code which supplies enough information for the Alto or Dolphin to read the next block of code, which is found on the second page of the boot file. From there, an EFTP connection serves to transfer the rest of the boot file from the server to the Alto or Dolphin, where packets are copied into main memory.

Even with the limited program space in the Alto bootstrap, it is possible to select which boot file you wish loaded. When the bootstrap starts, it reads a value from the unencoded keyboard, and places that value in the ID field of the MayDay Pup. Certain keyboard combinations are well known: holding down the BS key enables Ethernet booting; if BS is the only key held, boot file 0 (DMT, the dyanamic memory test) will be loaded; if BS and ' (Quote) are held down, boot file 10, the NetExec, will be loaded.

When booting a Dolphin, the mechanism becomes even more complex, as Alto-emulator microcode must be loaded. When the Dolphin has a working disk, this microcode is loaded from a set of special tracks shared between partitions, and not writeable during normal operations. When the Dolphin has no disk (or the disk hasn't warmed up yet), it tries to load microcode from the Ethernet. This *microcode boot* is similar to an Alto net boot, except that instead of using EFTP to move the microcode, the microcode boot server simply sends the code in Pups one after another, without waiting for acknowledgement.

The Tenex/Tops-20 Pup server job has the ability to perform both Alto-style and microcode-style boots, but cannot send *Breath-of-Life* packets (Tenex/Tops-20 Pup service can send only Pups). The VAX/Unix Pup service currently supports *Breath-of-Life* and Alto boot, and should be able to support microcode boot with minimal effort.

Chapter Four
The Interlisp-D Software

The Interlisp-D environment is comprised of a number of files. Chief among these are the Interlisp microcode loader, bootstrap, sysout, and virtual memory buffer. There are a number of ancillary files, such as Interlisp-D font files and Lispusers packages.

To run Interlisp, first boot the Dolphin. When the Alto emulator is running, and you're looking at the narrow Alto screen, run the program DolphinLispMC.Run. This loads the Interlisp microcode. As a result, the screen expands to its full width, and the Alto Executive returns. You may now start Interlisp-D. If Interlisp has been used before in this partition, the file Lisp.VirtualMem contains a runnable image of Interlisp, and by typing Lisp, you will restart this image. If Lisp.VirtualMem is empty (Lisp has never been run before, or you've just refreshed the disk), type Lisp/i. This causes the Lisp bootstrap (Lisp.run) to fill Lisp.VirtualMem with a runnable image of Interlisp taken from the Interlisp sysout file, Lisp.sysout.

Lisp takes a minute or so to load. When it's ready, you'll by sitting at Interlisp's top level, EVALQT. To leave Interlisp, evaluate the function LOGOUT. The next time you start Lisp, you'll be returned to the Interlisp environment at the point you logged out.

Files Needed to Run Interlisp-D

The following is a list of files which must reside on your disk to run Interlisp-D:

DolphinLispMC.Run
: A runnable program in Alto operating system format which loads the Interlisp microcode.

Init.Lisp
: Interlisp-D configuration file. Defines the search path for Lispusers directories, user greetfiles directories, etc. Read when the GREET function is evaluated. This file is comparable to the site Init.Lisp in Interlisp-10.

Lisp.Run
: The Interlisp-D bootstrap program, again an Alto format runnable file. If you type Lisp to the Alto Executive, Interlisp-D will be started from the environment stored in Lisp.VirtualMem. This is the way to continue an Interlisp session ended by LOGOUT.

  If you type Lisp/i to the Alto Executive, the Interlisp environment will be loaded anew from Lisp.sysout. This is the way to reinitialize Interlisp-D. This takes a bit of time, however, so be patient. When the hourglass cursor reaches the bottom of the screen, initialization is complete.

  If you type Lisp *filename*, the Interlisp environment is loaded from *filename* a file produced by the Interlisp-D SYSOUT function. Note: the previous Interlisp environment stored in Lisp.VirtualMem will be lost. If your fileserver has a 3 Megabit Ethernet BSP-based FTP server, you can load a sysout over the Ethernet by typing Lisp *{hostname}filename*, where *hostname* is your file server's name on the Ethernet.

Lisp.Syms
: An Alto-format symbol file for Lisp.Run. If your Dolphin falls into Swat, you may be able to continue running Lisp by calling RAID from Swat. To do so,

type Raid↑C (Raid followed by ↑C) to Swat. The first time you enter Swat, however, Lisp's symbols may be unknown. To make them known to Swat, type ↑YLisp.Syms).

Lisp.Sysout — The basic Interlisp environment, used to initialize Lisp.VirtualMem.

Lisp.VirtualMem — The virtual memory buffer. This file should be 8000 decimal pages long, and works best when allocated contiguously. The Createfile program, mentioned above, can be used to create Lisp.VirtualMem. To do so, type **Createfile Lisp.VirtualMem 8000D** to the Alto Executive. To ensure the availability of 8000 contiguous pages, this file is best created after refreshing the disk.

Files which must be resident on disk or accessible via a file server:

*.DCOM — Compiled Lispusers packages.

*.Strike — Interlisp-D font files (Gacha and Helvetica fonts).

Fonts.Widths — Defines font sizes for Xerox fonts.

Configuring Interlisp-D

When Interlisp-D is started up for the first time (after reinitialization, for example), it evaluates the GREET function. This causes it to read the file Init.Lisp on the disk. Figure 4-1 is a sample disk INIT.LISP.

```
(* This is the Stanford system greeting file.)
(VARS (DEFAULTPRINTINGHOST (QUOTE TAHOE))
      (LISPUSERSDIRECTORIES (QUOTE ({LASSEN}<LISPUSERS>
                                    {LASSEN}<DOLPHIN>LISPUSERS>
                                    {SUMEX-AIM}<DOLPHIN>
                                    {DSK})))
      (FONTDIRECTORIES (QUOTE ({DSK} {LASSEN}<ALTOFONTS>)))
      (USERGREETFILES (QUOTE (({LASSEN}< USER >LISP>INIT.DCOM)
                              ({LASSEN}< USER >LISP>INIT)
                              ({LASSEN}< USER >INIT.DCOM)
                              ({LASSEN}< USER >INIT.LISP)
                              ({SUMEX-AIM}< USER >INIT.DCOM)
                              ({SUMEX-AIM}< USER >INIT.LISP)
                              ({DSK}INIT. USER]
      (FONTWIDTHSFILES (QUOTE ({DSK}FONTS.WIDTHS
                               {LASSEN}<FONTS>FONTS.WIDTHS)))
      (DIRECTORIES (QUOTE (NIL {LASSEN}<LISPUSERS>
                               {LASSEN}<LISPUSERS>LISP>
                               {LASSEN}<DOLPHIN>LISPUSERS>
                               {LASSEN}<DOLPHIN>LISP>
                               {SUMEX-AIM}<DOLPHIN>
                               {DSK})))
      (ALTNARROWBACKGROUNDSHADE 42330))))
STOP
```

Figure 4-1: A sample disk resident INIT.LISP.

This file defines the default printer for LISTFILES1 to be TAHOE (Stanford's Dover printer), and then defines a number of directory search paths. LISPUSERSDIRECTORIES is searched when a FILES command from a COMS list specifies another file to load (i.e. `FILES FROM VALUEOF LISPUSERSDIRECTORIES ...`). FONTDIRECTORIES is searched looking for Strike files.

USERGREETFILES is searched looking for the user's personal init file (see below). FONTWIDTHSFILES is searched looking for Fonts.Widths. Finally, DIRECTORIES is searched by the FINDFILE function. ALTNARROWBACKGROUNDSHADE is an alternate texture (see the *Interlisp-D Users Guide*) for the background of the Interlisp-D windows system.

The GREET function also looks up the user's personal init file. The list USERGREETFILES defines where to look for this file. PACK* is used to build each candidate filename, with USER bound to the value of USERNAME. The Dolphin sets USERNAME by looking into the Alto OS. Username is initially the name specified when the operating system was installed (with the Install.~ command). The name can be changed by giving the Login.~ command. Figure 4-2 shows a typical personal init file.

The important thing to note in personal init files is that they can be used in many different Interlisp environments. `(SELECTQ (SYSTEMTYPE) ... )` is used to choose options based on the implementation of Interlisp being run. In this file, for instance, when running Interlisp-10, <XMYCIN>TTYIN.COM is loaded, and all control characters (except ↑Y) are made separators.[7] When running Interlisp-D, the TTYIN, display editor, and data inspector packages are loaded, and the window system is turned white on a black background.

[7] The latter allows prettyprinted output with font change characters embedded to be read. Interlisp-D produces such files by default with MAKEFILE.

```
(E (RESETSAVE CHANGECHAR NIL))
   (DECLARE: FIRST
        (P (SELECTQ (SYSTEMTYPE)
                    ((TENEX TOPS20)
                     (SETSEPR (QUOTE (%| 1 2 3 4 5 6 7 8 9 10
                                         11 12 13 14 15 16 17 18
                                         19 20 21 22 23 24 26))
                              1 FILERDTBL)
                             (JSYS 34 64)
                     (SELECTQ (PEEKC T)
                             ((%  %↑J)
                              (CLOSEF (INPUT))
                              (SETQ USERNAME NIL)
                              (GREET0)
                              (RESET))
                             NIL)
                     (LOAD? (QUOTE <XMYCIN>TTYIN.COM)
                            (QUOTE SYSLOAD)))
                    (D (SETQQ USERNAME SCHOEN)
                       (VIDEOCOLOR T)
                       (CHANGEBACKGROUND WHITESHADE)
                       (LOAD? (QUOTE TTYIN.DCOM)
                              (QUOTE SYSLOAD))
                       (LOAD? (QUOTE DEDIT.DCOM)
                              (QUOTE SYSLOAD))
                       (LOAD? (QUOTE INSPECT.DCOM)
                              (QUOTE SYSLOAD)))
                    (PRIN1 "INIT.LISP: Unknown system type"))))
        (VARS #RPARS PROMPT#FLG (FIRSTNAME "Eric")
              (CLEANUPOPTIONS (QUOTE (RC)))
              (CLISPIFYENGLSHFLG NIL)
              (HARDCOPYFLG)
              (ADISOFILE)
              (DISPLAYON)
              (MSPRINTFLG 1)
              (CHECKUNSAVEFLG T)
              (COMPILEMODE (COND ((BOUNDP (QUOTE COMPILEMODE))
                                  COMPILEMODE)
                                 (T (QUOTE PDP-10))))
              (ALTO NIL)
              (EMACSFLG T)
              (SHOWPARENFLG T))
        (APPENDVARS (DIRECTORIES NIL {LASSEN}<LISPUSERS>
                                 {SUMEX-AIM}<LISP>)
                    (LISPUSERSDIRECTORIES {LASSEN}<LISPUSERS>
                                          {SUMEX-AIM}<LISP>))
        (ALISTS (FONTDEFS PARC))
        (ADDVARS (INITIALSLST (SCHOEN . ejs:)))
        (P (FONTSET (QUOTE PARC]
STOP
```

Figure 4-2: A typical personal Interlisp init file.

I. Reinstalling a Disk Partition

The long form of the Install.~ command allows you to completely erase and rewrite a disk partition. **You must only attempt this if you have access to 3 Megabit Ethernet software!** The following procedure will completely erase your disk partition, and then rewrite the Alto OS, Executive, and FTP program. The Alto files will be FTP'ed from the host you specify in the procedure. This can be a large file server, such as an IFS or timesharing machine running PUP software, or it can be another Dolphin. If your server for this procedure is a Dolphin, then enter a blank line in the question below asking for the name of the directory where Alto files are kept. In the example below, it is assumed Alto files are kept in a directory named *Alto*. **Make absolutely sure all important files are safely stored on another machine!**

The disk refresh procedure requires a server supporting the Alto boot protocol be running somewhere on your Ethernet. Some file servers, such as the Sumex-Tenex and DECsystem-20 PUPSRV program, and the Xerox IFS support the Alto boot protocols. If your file server doesn't, and you have at least one running Dolphin, you can turn that Dolphin into a boot server by running the Peek program on it. Refer to page 16.

Before reinstalling a disk partition, you may run an overnight disk exercisor. This is wise if you suspect hardware problems within the disk. The instructions immediately following this paragraph tell you how to run the disk exercising program. If you believe the disk is sound, skip them, and proceed to step 2 for instructions on rewriting a disk partition.

1. To run the exercisor, you need a boot server with BFSTest.boot. From the Alto executive on the machine you wish to exercise, in the proper partition, type NetExec to boot the NetExec program. Then type BFSTest to invoke the disk exercisor. Do the following:

```
*Partition 1 (or 2, whichever you want done)
*Certify↵ How many passes? 100
 Use both of the disks? Y
 Use all 406 cylinders of the disk? Y
 Use all 14 sectors of the disk? Y
```

   Note: It seems Certify sometimes does not give you the option of 406 cylinders, and assumes 203 instead. If this occurs, run NewOs as described below first, and then run BFSTest.

```
*Erase
 Use both of the disks? Y
 Use all 406 cylinders of the disk? Y
 Use all 14 sectors of the disk? Y
*Exercise↵ How many passes? 10
*Quit
```

2. Quiting returns you to the NetExec program. Now reinstall the software on the disk by running the NewOS program. If you have not just exercised the disk with BFSTest, and are instead just reinstalling the software, get to the Alto Executive and type Install.

3. Respond to the following dialog as follows:

```
Do you want the long installation dialog?  Y
Do you want to ERASE the disk . . .?  Y
Type the name of a host to get Alto programs from: hostname↵
Type directory name where Alto files are kept: Alto↵
Include DP1 in file system?  Y
Use all 14 sectors of the disk?  Y
Do you want a big SysDir?  Y
When the disk is ready type OK to proceed, A to abort.  OK↵
```

Take a short coffee break, and pray that you have previously saved all your important files somewhere else.

```
Do you want to disable error logging through the net? Y
What is your name?  Whatever you want for your default name↵
Please give your disk a name?  Partition 1 or Partition 2↵
Do you wish to give your disk a password? Y or N[8]
```

If you answered yes, then

```
Please type the password: Choose your password↵
```

At this point in time the machine, should connect with the host and retrieve Executive.Run and Ftp.Run. It will then return you to the Executive with a clean disk.

4. Now retrieve InstallSwat.Run and Sys.Errors. Then run InstallSwat.Run. After it is done delete InstallSwat.Run, Dmt.Boot, and Dumper.Boot. Next retrieve CreateFile.Run and type the following to the Exec: Createfile Lisp.VirtualMem 8000d (takes about 5 min.)

5. Next bring over the appropriate lisp files and anything else you might need. The following procedure is the one used within Stanford to refresh Dolphins, and reflects the network topology and directory structure within the Computer Science Department and the SUMEX-AIM project. These retrieves are best done with the Alto FTP program (FTP.Run). A newly refreshed disk will have a runnable FTP program.

   a. Retrieve Lisp.Run, Lisp.Syms, Lisp.Sysout, and DolphinLispMC.Run from {IFS}<Dolphin>lisp> (i.e. file server host IFS, directory Lispusers, subdirectory lisp). If the IFS isn't available, these files can also be retrieved from {SUMEX-AIM}<DOLPHIN>.

   b. Retrieve *.STRIKE and INIT.LISP from {IFS}<Lispusers>.

   c. Confirm that Lisp runs by initializing it via Lisp/i.

If you need to create your own temporary boot server, you can do run the Alto Peek program on another, working, Dolphin. There are two files you need in addition to the boot files. These are Peek.run and PeekUser.cm. You should rename the file User.cm to some other file name (i.e. Rename.~ OUser.cm ← User.cm), then copy PeekUser.cm to be User.cm. Then run the Peek

---

[8] Don't type a carriage return! Install doesn't wait for your carriage return, and if you do type one, you will have ended up supplying a null password for the disk, which is a fatal mistake, as you will never be able to log in. If by mistake you end up supplying a null password, you'll have to reinstall the operating system via the NetExec and NewOS; this time, don't erase the disk.

program. The Dolphin running the server program must have the necessary boot files on its disk. For the above procedures, you need NewOS.Boot and BFSTest.Boot. When you are done, rename OUser.cm back to User.cm.

II. Recovering From Disasters

Sooner or later, someone is going to delete the wrong file, or the disk will develop a few hard read errors in the wrong places, or something similar, and your Dolphin will be unusable. Typically, you'll boot the machine, the maintenance panel will flash the standard disk boot sequence (40-700-720-etc), but when the Alto-sized display lights up, the cursor won't track with the mouse. This indicates that the Alto software is hung somewhere due to a bad or missing file. Unless you have access to a file server with complete booting service (Breath-of-Life and net boot), call Xerox. If you do have booting software on your file server, there is still hope.

The first program to run is Scavenger (the boot file version, of course). To do this, you must first run the NetExec program. Since you can't run the Alto Executive to give the NetExec.~ command, you must get NetExec in another way: hold down simultaneously the BS and ' (Quote) keys, and press the boot button (try first the one behind the keyboard, and if that fails, the START/ON button on the processor's front panel). Continue holding the keys until the Alto-sized screen appears, with a small randomly-filled square sitting in the center. This square represents the acknowledgement packet for the EFTP transaction taking place to boot the Dolphin (with NetExec.boot); it is stored in the cursor memory to save on critically-short memory space during booting. In short order, the booting process should end, and the NetExec program should start. Select the faulty partition with the NetExec's Partition.~ command, and type Scavenger↵, and the booting process should repeat itself, completing this time with the Scavenger program running. Answer Scavenger's questions and cross your fingers. If it wants to rewrite a page, let it. On its completion, Scavenger will attempt to boot into the current disk partition.

If the Dolphin still won't boot, try reinstalling the operating system. To do this, obtain the NetExec as described in the previous paragraph, and select the desired disk partition. Type NewOS↵, and wait for the booting to end. Answer questions as in the following dialogue:

```
Do you want to install this operating system? Y
Dov you want the long installation dialogue? N
```

Follow the instructions in Appendix I to complete the installation. When NewOS finishes, it should have supplied you with a new operating system, default font, and Executive. You may have to repeat the entire procedure a couple of times, in case the disk surface is badly damaged, and NewOS has trouble finding good pages to install itself on. If you still can't get the Dolphin to boot, reinstall the faulty partition, following the instructions in Appendix I; you probably should also check the condition of the disk with either DiEx (page 9) or BFSTest.

III. Maintenance Panel Codes

The following is a list of maintenance panel codes transcribed from some unnamed Xerox document. Complete accuracy is not guaranteed.

MP Codes from Rev L EPROMS

0000	One of the first few instructions in the EPROM clears the MP.
0001	The first few ALU tests worked OK. You should never see this, since some other error should happen or the MP should change to 0040 when RDC booting starts.
0002	Midas boot (Midas is the hardware debugger).
0004 to 0015	One of the preliminary ALU tests failed.
0016	Mismatch after write then read of an R register via the stack.
0017	The contents of an R register have changed.
0020 to 0035	A fault happened. The MP contains 20 plus the contents of the Parity register. A fault in the fault handler will reBoot the machine, so you may not get to see these codes.
0020	A Breakpoint micro instruction (F1 = Group, B, F2 = 7) was executed.
0021	Memory error. Since the EPROM code doesn't use main memory, this is probably an H4 Parity error.
0022	R register parity error.
0024	Control Store parity error.

0022 and 0024 are to be expected if you have just powered up your machine (The bias on the RAM chips hasn't been pumped up yet). This will invoke a one minute delay to avoid hogging the boot servers when something is broken. Poke the button again if you want faster service.

0028	Stack error.
0040	Starting to load microcode from RDC.
0041	Can't find RDC (Will now try to EtherBoot).
0042	Hardware error while reading.
0043	Seek timed out.
0044	Microcode checksum error.
0045	Bad Control Store address -- attempt to load into EPROM area.
0046	Disk not ready (Will now try to EtherBoot).
0047	The label word which should contain a link to the next page of microcode to be loaded has an invalid disk address.

Most disk errors (0042, 0043, 0044, 0045, 0047) can be caused by simple transient read problems. The RDC task simply retries all of them while the emulator task is counting down a timer. If the timer runs out, you will see 0048.

0048	Didn't load microcode from RDC within one second (Will now try to EtherBoot).

0060 Starting to load microcode via Ethernet.

0061 Can't find Ethernet board.

0062 Bad microcode checksum.

0063 Bad Control Store address -- attempt to load into EPROM area.

0064 Hardware error (bad status) at end of packet.

0065 Timeout after 15 tries at about 10 seconds each.

If EtherBooting doesn't work, the MP will slowly alternate between 006x and 004x so that you can see both what is wrong with the disk and what is wrong with the Ethernet.

0070 to 0085 An unexpected wakeup happened. The MP contains 70 plus the number of the offending task.

0071 Unexpected wakeup for task 1.

0072 Unexpected wakeup for task 2.

0073 Unexpected wakeup for task 3.

0074 Unexpected wakeup for task 4. (Task 4 is used by the RDC and for Ethernet output.)

0075 Unexpected wakeup for task 5.

0076 Unexpected wakeup for task 6.

0077 Unexpected wakeup for task 7.

0078 Unexpected wakeup for task 10B. (Task 10B is used for Ethernet input).

0079 Unexpected wakeup for task 11B.

0080 Unexpected wakeup for task 12B.

0081 Unexpected wakeup for task 13B.

0082 Unexpected wakeup for task 14B.

0083 Unexpected wakeup for task 15B.

0084 Unexpected wakeup for task 16B.

0085 Unexpected wakeup for task 17B.

If you get one of these, one of the IO controllers is probably broken. For example, its reset logic is not working, or the wakeup logic on the RDC or Ethernet board is generating the wrong task number.

MP Codes from the (Oct 80) CSL microcode (Alto mode)

0100 Start Map Init.

0101 Not enough memory (You need 192K = 2 boards).

0102 Bad map.

0110 Start Device Init.

0115 Start Ethernet Init.

This is what you will see during and after booting the net exec from the Ethernet.

0120 Start Disk Boot.

This is what you will see after you have booted from the disk.

0121	Timeout waiting for disk status.
0122	Hardware error reading disk (after 10 retries), or ALU is too hot.
0128	R parity error.
0129	Real breakpoint.
0130	Map out of bounds.
0131	H4 parity error (The microcode ignores these if they come from task 7, Ethernet input, unless you have set a flag from Midas).
0132	H4 parity error and Map out of bounds.
0133	MC2 error.
0134	2 MC2 errors.
0135	MC1 Fault when emulator not ready.
0136	Fault from instruction following LoadPage.
0137	Stack Error (when emulator not ready).
0138	Control Store parity error.

MP Codes from Mokolumne Initial

0700	Start Map Init.
0701	Not enough memory (Initial requires 64K).
0702	Bad Map.
0720	Starting to load emulator microcode from disk.
0721	No RDC.
0722	RDC Read error.
0725	Starting to boot Pilot.
0728	R or CS parity error.
0729	Real breakpoint.
0737	Stack over/underflow.
0739	Generic memory error.
0740	Starting to EtherBoot.
0741	Can't find UTVFC (or Tor display controller).

*When Initial starts Etherbooting, it puts 740+2*bfn into the MP (BFN is the microcode boot file number). If EtherLoad can't load that file within a reasonable length of time, it will give up and bump the MP by one).*

0742 and 0744 are from the Amargosa version of initial.

0758	Trying to load AltoLF.eb
0759	Timeout trying to load AltoLF.eb
0760	Trying to load AltoCSL.eb
0761	Timeout trying to load AltoCSL.eb
0809	Pilot microcode not installed on SA4000 (Try to EtherBoot).
0810	Germ not installed on SA4000 (Try to EtherBoot).
0811	Physical Boot Volume not set on SA4000 (Try to EtherBoot).

0812 Label check from SA4000 (Try to EtherBoot).

MP Codes from Mokolumne (Pilot) Microcode

0140 Running Pilot (Tor kludge).

0710 Start device initialization.

0727 R register parity error.

0728 CS parity error.

0729 Real breakpoint.

0730 Map out of bounds.

0731 H4 parity error (The microcode ignores these unless you have set a flag from Midas).

0732 H4 parity error and Map out of bounds.

0733 MC2 error.

0734 2 MC2 errors.

0735 MC1 fault when emulator not ready.

0736 Fault from instruction following LoadPage.

0737 Stack error (when emulator not ready).

0846 Running Pilot (this was the last value before the Tor kludge was added). You won't normally be able to see this since the Germ smashes the MP to 0900 as soon as it gets a chance.

MP Codes from the Hardware

8808 R parity error.

8880 CS parity error.

8888 Lamp test (while the boot button is down).

Index